Building Complex Objects
(Without Code)

Dennis Pegden
Simio LLC
Outline

- What is an Object and Why Build Custom Objects?
- Building a Hierarchical Object
- Building a Base Object
- Sub-classing an Object
Simio Object View

- Models the physical objects in the system.
- Objects combine data and functionality into self-contained units.
- Objects serve as a model of an abstract "actor" that can perform work, report on and change its state, and "communicate" with other objects.
- New objects may be built from other objects or from graphical processes.
Process View

- Models the flow of entities (tokens) through a series of process steps.
- Process steps can take place over time.
  - Seize – Delay - Release
- Can be used to create new objects.
- Can be used to customize objects without actually changing the object’s definition.
The Simio Advantage

- Even new users can quickly create custom objects, **without writing code**
- Objects can easily be shared between projects and users
- Intelligent objects can be aware of and interact with each other and make smart decisions
- The logic of the objects in our Standard Library is exposed and can be used as a basis for a new, custom object
Why Build Custom Objects?

- **Utility objects**
  - Simple, reusable

- **Complex objects**
  - Custom data and behavior

- **Share expertise among broader user group**
  - Custom libraries might be easier to learn

- **Build hierarchical models**
  1. Create a model of a workstation
  2. Place three workstations into a work cell
  3. Place five work cells into a model of the facility
The **Crane Library** is a collection of objects that are specifically designed to work together in order to model one or more cranes operating simultaneously in a bay.

The Crane Library is not currently part of the standard Simio installation but may be easily downloaded for free. To download, go to the **Support** tab in the Simio software and click on the **Shared Items** button.
The **Transportation Library** is used to model **trucks, ships, trains, robots, and pipes.** The library works with both discrete and flow entities.

The Transportation Library is not part of the standard Simio installation but may be easily downloaded for free. To download, go to the **Support** tab in the Simio software and click on the **Shared Items** button.
Projects and Libraries

- A **project** may contain just one model or multiple models.
- A **library** is a collection of objects (models).
- Opening a project file verses loading a project as a library:
  - Opening a project file exposes the object definitions
  - Load a project as a library to use the objects in a model
Building Object Definitions

Base
(built from processes)

Derived
(subclassed from another object)

Hierarchical
(contains objects, and/or other models)
Anatomy of an Object

Object behavior is defined by a model built using processes and/or objects.

Associated nodes provide entry/exit to the object.

Graphical representation for the object.

Attached queues animate queue states.

Static inputs to the model logic.

Dynamic values that change during the run.

Properties

States/Events
Object Properties

Properties are the inputs into objects.

An object cannot “see” anything outside of itself so a property is used to send information into an object.

- **Examples:**
  - An Expression property might be used so that the user can send in a processing time for an object.
  - An Event property can be used to send the name of an event to the object so it can react when that event occurs.
  - An Object property might be used so an object is aware of another object in the model.

When building an object, consider what outside information this object will need to access.
Properties are assigned to a Category, which may be initially collapsed or expanded.

Properties may be assigned to a Switch that is controlled by a condition.

The Display Name may have blanks and special characters.

Properties have a Default Value.

Properties may have a Required Value.

Properties may be non-editable or Invisible.

The property Description is displayed when the user selects the property.
Tandem Server

Server 1
- Starved
- Processing
- Blocked
- Failed
- OffShift

Server 2
- Starved
- Processing
- Blocked
- Failed
- OffShift

Process Logic
- Process Time One
- Process Time Two

Buffer Capacity
- Input Buffer
- Output Buffer

Zero-time transfer, no buffers.
Base Lathe

Lathe

InputBuffer  Processing  OutputBuffer

Input  Output

Process Logic
- Transfer In Time
- Processing Time

Buffer Capacity
- Input Buffer
- Output Buffer

Enters the InputBuffer and waits to transfer to Processing.

Enters Processing, delays by Processing Time, and waits to transfer to OutputBuffer.

Enters OutputBuffer, and waits to transfer to Output node.
Entry/Exit via Nodes

Process Logic

External View

Input Logic Type is Process Station

This process is triggered by the station Entered event.

The token's associated object is the arriving entity.

Transfer out from the exit node is done using the Transfer step

Object Logic

External View

Input Logic Type is Facility Node

The arriving entity is sent to the Facility Node that is specified.

Transfer out from the exit node is done using Auto Transfer On Entry

Object

www.simio.com | Copyright 2013 Simio LLC | All rights reserved.
Sub-Classing

Inherit - Change - Extend

Overrides the Base Object process to change behavior.

Inherits this process from the Base Object, therefore it cannot be edited.

Adds this as a new process to provide additional logic.
Sub-classing Library Objects

- The Standard Library objects are completely open.
- Drag library objects into your project to create a sub-classed object.
- Edit the sub-classed object.
  - Override processes.
  - Add new processes.
  - Add graphics to the External view.
  - Hide/Rename inherited properties.
  - Add new properties.
Server with Repair Person

Override this inherited process and add a Seize and Release step.

Switch in if Failure Type != NoFailure.

Rename Processing Time

Process Logic
Treatment Time

Reliability Logic
Repair Person
Any model can provide the logic for an object definition – just add an interface and external view

Object definitions can be built three ways:
- **Base** objects using processes.
- **Derived** objects using inheritance.
- **Hierarchical** objects using composition.

Custom objects can be created by new users very quickly